**基于RNN的古诗续写、藏头诗生成器**

**原因：**

学长分享了RNNLM demo——诗的续写后，我对这个项目非常感兴趣。在本次期末大作业中，我选择做一个诗的续写项目。在原有的RNNLM demo上进行修改，完善诗歌续写的功能，并添加藏头诗的功能。

**实验实现过程：**

1. 数据预处理
2. 语料准备：四万多首古诗

https://github.com/AaronJny/poetry\_generate/blob/master/origin\_data/poetry.txt

1. 语料预处理：将文字转换为机器能理解的形式

具体过程：创建分词器，将词和编号一一对应、编码和解码操作，并添加起始标记CLS及结束标记SEP；加载数据集，用冒号统一成相同格式；处理数据：冒号分割标题、不能含有禁止词、不能超过最大长度；统计词频：过滤低频词、按词频排序、去掉词频，保留词列表；将特殊词和数据集中的词拼接；建立映射关系、使用新词典重新建立分词器、混洗数据；数据集生成器：导入数据；padding填充（不足就填充 超过就截断）；创建一个用于训练的生成器。

1. 模型

词嵌入——LSTM模型——配置优化器和损失函数

LSTM模型——第一层 128 dropout=0.5 第二层128 dropout=0.5

每一个时间点的输出都做softmax处理以便预测下一个词的概率

1. 训练模型

每个epoch训练完成后，保留最优权重（初始loss赋予很大的值进行迭代）。开始训练，并在训练过程的每个epoch迭代中采样。

1. 验证模型

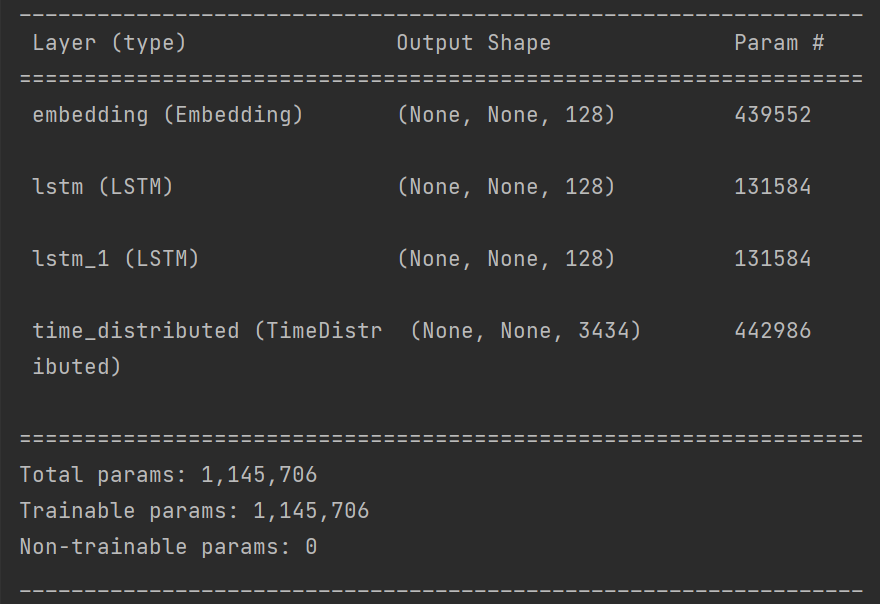
两种方法

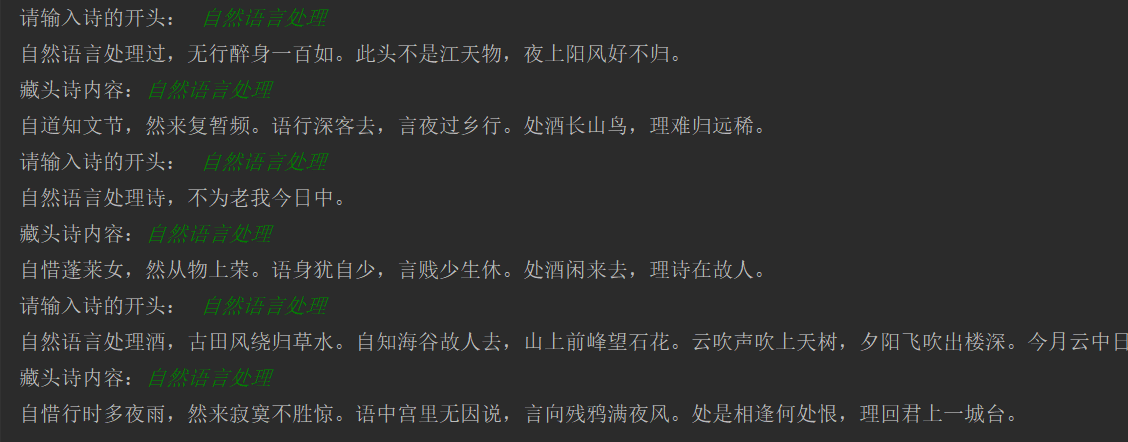
1. 续写：转成token 进行预测，按照出现频率对所有token倒序排列，排列后对概率进行归一，按预测概率随机选择词作为预测结果，保存。
2. 藏头诗：空串初始化；藏头操作：记录下藏头诗的第一个字加入到列表中开始生成一个短句；进行预测：只保留第一个样例，按出现概率对所有token倒序排列，对概率归一，按预测的概率随机选择一个词作为预测结果，保存。

**实验环境：**

tensorflow ==2.10.0  
 numpy==1.23.4

**结果展示：**





**写在最后：**

感谢本学期几位老师和学长的教导带我走进NLP世界的大门，课程虽然结束，但学习不会因此而止。仍然记得朱老师最后一堂课上的分享，“有希望的坚持不是一件难事，但很多事是坚持了才会有希望”，这句话不仅在我的学习上带来一定的启发，在我的生活上更是受益良多。最后，祝愿老师们身体健康，祝学长们早日毕业拿offer！